Lógica para la Computación

Convocatoria de julio, 14/07/21

Nombre: DNI:

NOTA: Es necesario un mínimo de 3 ptos (el 50% de la puntuación total) en la prueba para sumar las prácticas correspondientes. La duración del examen es de 2 horas.

1. (1.5 ptos) Sea el predicado `hanoi(Num,A,B,C,Movs)` definido por las cláusulas

```
:- op(600,yfx,a).

hanoi(1,A,B,_,[A a B]):-!.
hanoi(N,A,B,C,Movs) :- N1 is N - 1,
                       hanoi(N1,A,C,B,Movs_1),
                       asserta((hanoi(N1,A,C,B,Movs_1):-!)),
                       hanoi(N1,C,B,A,Movs_2),
                       retract((hanoi(N1,A,C,B,Movs_1):-!)),
                       append(Movs_1,[A a B|Movs_2],Movs).
```

que calcula los movimientos `Movs` que resuelven el problema de las *Torres de Hanoi* para el desplazamiento de `Num` discos desde el palo `A` al `B`, tomando el `C` como intermedio. Supongamos que realizamos la pregunta:

```
:- hanoi(4,a,b,c,Movs).
```

explicar razonadamente si el predicado `asserta/1` (resp. `retract`/1) contribuye o no, en este caso, a evitar la multiplicación de cálculos (resp. la acumulación de reglas aplicables a una misma pregunta).

El predicado `asserta/1` *no cumple, en este caso, con la función comentada. La razón es que, cuando se realiza la inclusión de cláusulas mediante* `asserta/1`*, se hace sobre valores constantes para las variables* `A`*,* `B` *y* `C`*.*

Así, para la pregunta planteada, la primera llamada a `asserta/1` *en la segunda cláusula del predicado* `hanoi` *se corresponde a* `asserta((hanoi(3,a,c,b,Movs_1):-!))`*, una vez resuelta la pregunta* `hanoi(3,a,c,b,Movs_1)`*.*

El objetivo sería evitar la duplicación de cálculos asociada a la llamada `hanoi(3,c,b,a,Movs_2)` *que seguiría al* `asserta((hanoi(3,a,c,b,Movs_1):-!))`*, pero tal cosa no es posible porque la cabeza de la cláusula*

`hanoi(3,a,c,b,Movs_1):-!.`

no unifica con ese segundo término `hanoi/5`*, que no es otro que*

`hanoi(3,c,b,a,Movs_2)`

En cuanto al predicado `retract/1` *si cumple en este caso con la función comentada. Así, continuando con nuestro ejemplo, la primera llamada a* `retract/1` *en la segunda cláusula del predicado* `hanoi` *se corresponde a* `retract((hanoi(3,a,c,b,Movs_1):-!))` *que, en efecto, elimina la cláusula*

`hanoi(3,a,c,b,Movs_1):-!.`

que previamente habíamos introducido y que resulta inservible para la resolución de la pregunta planteada.

2. (1.5 ptos) Describir un programa PROLOG que presente ciclicidades en la resolución. Plantear al menos dos soluciones razonadas a dicho problema en ese caso concreto, ilustrándolas mediante árboles de resolución.

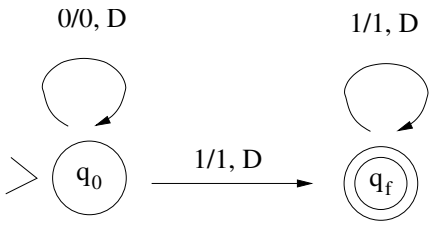
Basta con considerar el programa planteado en el Ejemplo 37 (pag. 13) comentado en el texto accesible en el enlace

`Documentos y Enlaces > Material de estudio > Prolog > prologIA.pdf`

de la entrada Moovi de la asignatura, donde ya se plantean dos posibles soluciones (podrían aplicarse otras alternativas) al problema de los ciclos.

3. (1.5 ptos) Describir razonadamente y mediante un grafo una Máquina de Turing capaz de reconocer el lenguaje $\mathcal{L} = \{0^n 1^m, \text{ con } n \geq 0, \ m \geq 1\}$. Trazar sus movimientos para la entrada $w = 001$.

El grafo asociado a una de las posibles TMs que reconocen $\mathcal{L}$ es



La traza para la entrada 001 sería entonces la siguiente:

$$q_0001 \vdash 0q_001 \vdash 00q_01 \vdash 001q_f$$

4. (1.5 ptos) Dado el predicado PROLOG `member/2`, verificando la pertenencia de un elemento `E` a la lista `L`:

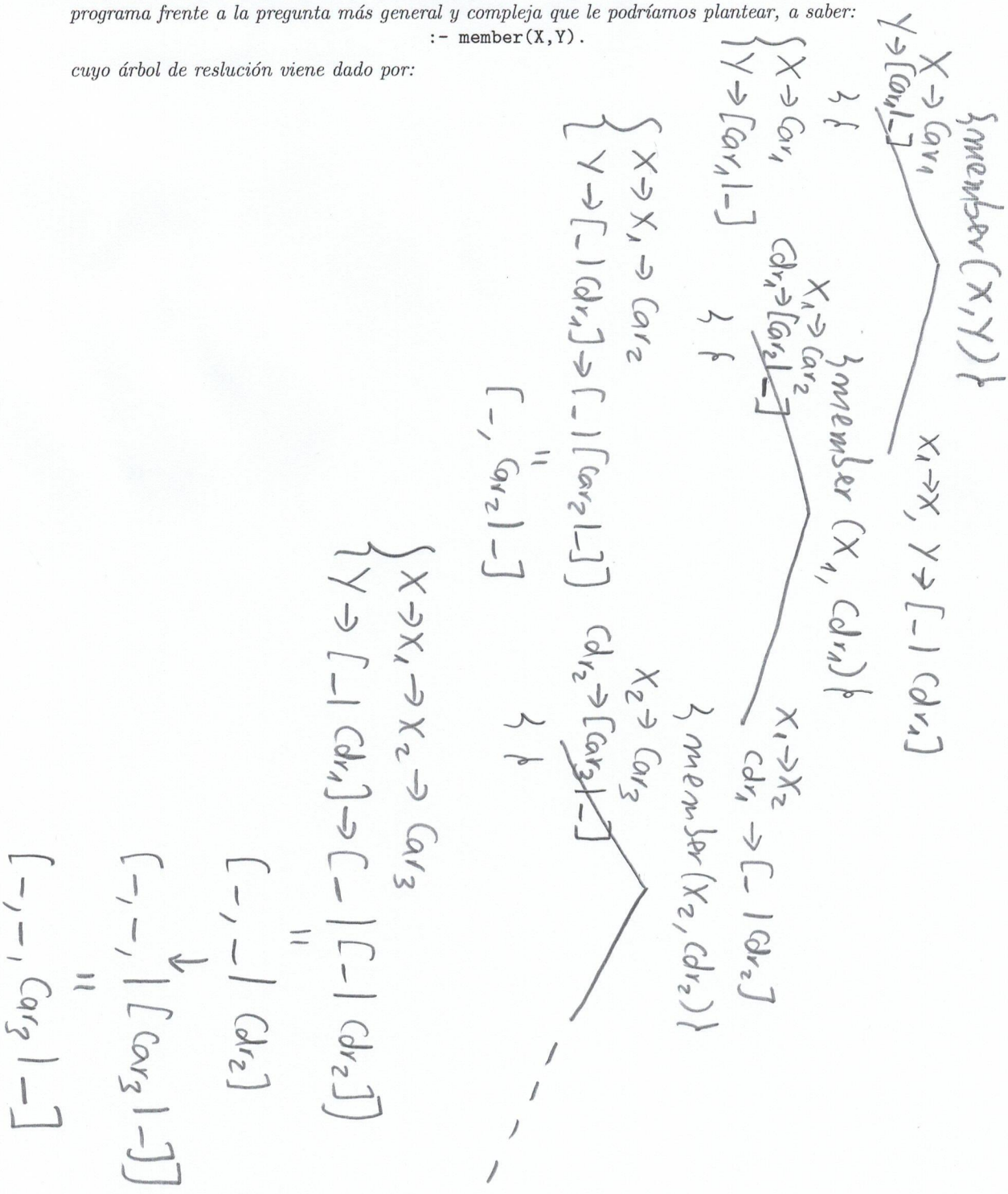
```
member(Car,[Car|_]).
member(X,[_|Cdr]):-member(X,Cdr).
```

explicar razonadamente si el programa es completo y si presenta problemas de sesgo en sus respuestas.

No hay problemas de sesgo[1]*., pero si de completud, dado que no siempre es posible generar el conjunto completo de soluciones en tiempo finito. Para demostrarlo bastará con observar el comportamiento del programa frente a la pregunta más general y compleja que le podríamos plantear, a saber:*

```
:- member(X,Y).
```

cuyo árbol de reslución viene dado por:



[1]El programa es capaz de generar todas las respuestas posibles.